UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/092,168	03/06/2002	Sridhar Satuloori	5681-08800	9232

7590 08/22/2007

Robert C. Kowert
Conley, Rose, & Tayon, P.C.
P.O. Box 398
Austin, TX 78767

EXAMINER	
ZHEN, LI B	

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
08/22/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

d9

Office Action Summary	Application No.	Applicant(s)	
	10/092,168	SATULOORI ET AL.	
	Examiner	Art Unit	
	Li B. Zhen	2194	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *29 May 2007*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-14 and 16-53* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-12,14,16-25,27-39 and 41-52* is/are rejected.

7)☒ Claim(s) *13,26,40 and 53* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.	6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1 – 14 and 16 – 53 are pending in the application.

Response to Arguments

2. Applicant's arguments filed 05/29/2007 have been fully considered but they are not persuasive. In response to the Non-Final Office Action dated 02/28/2007, applicant argues:

(1) Lucassen teaches that new developer-generated interaction logic layers are used to generate new presentation layers. Thus, when a developer creates a new interaction logic layer, Lucassen's system will generate a new presentation layer accordingly. However, Lucassen's system does not include any dynamic component generator determining whether a new set of requirements for the application includes changes from an initial set of requirements for the application. Generating a new presentation layer based on a new interaction logic layer does not disclose or anticipate a dynamic component generator configured to determine whether a new set of requirements includes changes from an initial set of requirements, as recited in Applicants' claim. [p. 16];

(2) Green does not teach a dynamic component generator determining whether a new set of requirements for an existing application includes changes from an initial set of requirements for the application. [p. 16];

(3) Contrary to the Examiner's suggestion, the cited passages of Green do not describe receiving new requirements for an application that already exists, as

required by Applicants' claims, but receiving requirements for a new application. [p. 17]; and

(4) As described in paragraph [0060], it is the application development system, not an existing application, which is updated in response to receiving requirements for a new application. When Green discusses modifying existing software components at paragraph [0060], Green is referring to components in the application development system, not components of an existing application. [p. 18].

Examiner respectfully disagrees.

As to argument (1), it is noted that the previous office action relied on Lucassen to teach a dynamic component generator that is configured to receive a new set of requirements for the application and generate a second dynamic component to replace the first dynamic component, wherein the second dynamic component is configured to function according to the new set of requirements [see the rejection to claim 1 below]. The previous office action relied on the secondary reference (Green) to teach a dynamic component generator configured to determine whether a new set of requirements includes changes from an initial set of requirements. Lucassen discloses an authoring tool that allows users to edit a view of an application that results in an update of the interaction logic layer and creation of customization meta-data [i.e. p. 3, paragraph 0028]. Modifying the interaction logic and generating customization meta-data creates new requirements for the view of the application. The customization meta-data represents the new requirements for the view of the application and the interaction manager receives the interaction logic layer and the customization meta-data and generates a new view that reflect the changes made by the user [i.e. p. 12, paragraph 0109]. Therefore, Lucassen teaches a dynamic component generator configured to receive a new set of requirements for the application and generate a second dynamic component to replace the first dynamic component, wherein the second dynamic component is configured to function according to the new set of requirements.

In response to argument (2), it is noted that the features upon which applicant relies (i.e., an "existing application") are not recited in the rejected claim(s). Although the claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993).

As to argument (3), it is noted that the claims and specification do not recite or disclose an "existing application". Thus, the claims do not require receiving new requirements for an application that already exists. Applicant appears to assert that an application does not exist until after development of the application. Examiner disagrees and notes that an application can exist in the development stage. For example, an application has to exist in order for it to be tested during the development stage. While some of the claims recite installing application modules [i.e. claims 14 and 17], it is noted that during the development stage the application would also need to be installed in order to test the application. Green teaches that the system is in production [step 52, Fig. 5] and then new application requirements come in [step 60, Fig. 5]. When the system is in production, the application at least exists in the development stage. Thus, Green teaches receiving new application requirements for an application that already exists. Since the claims do not specify whether the application exists in the development stage or the post-development stage, the application can be interpreted as an existing application in the development stage. Therefore, the combination of Lucassen and Green teaches applicant's invention as claimed.

As to argument (4), Green teaches that the application system is in production [step 52, Fig. 5] and then new application requirements come in [step 60, Fig. 5]. When the application system is in production, the application at least exists in the development stage. The application system that is in production is periodically reviewed for adjustments that may be necessary [step 54, Fig. 5]. The software components and tiers of the application in production are modified or new components are created in response to the new application requirements [steps 62, 64 and 66, Fig. 5; p. 5, paragraph 0060]. After the software components are added or modified according to the new application requirements, the process returns to system production [see Fig. 5, arrows loop back to step 52] to update the application. Therefore, Green teaches modifying the software components of an existing application in the development stage.

Allowable Subject Matter

3. Claims 13, 26, 40 and 53 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten in independent form including all of the limitations of the base claim and any intervening claims.

Claim Rejections - 35 USC § 103

4. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

5. This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

6. **Claims 1-12, 14, 16-25, 27-39 and 41-52 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent Application Publication No. 2003/0023953 to Lucassen et al. [hereinafter Lucassen] in view of U.S. Patent Application Publication No. 2002/0104067 to Green et al. [hereinafter Green], both references cited in the previous office action.**

7. As to claim 1, Lucassen teaches the invention substantially as claimed including a system [MVC-based development system; p. 2, paragraph 0017], comprising:

a processor [p. 7, paragraph 0067];

a computer-accessible medium coupled to the processor, wherein the computer-accessible medium is configured to store program instructions executable by the processor [Tier-2 comprises the business logic that runs on a Web application server, Web server; p. 7, paragraph 0067; examiner notes that the MVC framework includes servers and a processor and memory to store program instructions are inherent to the servers] to implement an application program [a application 50, Fig. 5; p. 11, paragraph 0105] comprising:

one or more application modules [an application data layer 51, a business logic layer 52, an interaction logic layer 53 a customization layer 54, and application process 55; p. 11, paragraph 0105], wherein at least a first one of the application modules comprises a first dynamic component [interaction logic layer 53; p. 11 – 12, paragraph 0107] and a static component [application data layer 51 comprises data content, file services and databases, and comprises all of the backend information; p. 11, paragraph 0105], wherein the first dynamic component and the static component are configured to function according to an initial set of requirements for the application [ability of the system to use the best possible combination of interface modalities based on the user's current preferences, needs and abilities as well as the application requirements and device capabilities; pp. 4 – 5, paragraph 0041]; and

a dynamic component generator configured to receive a new set of requirements [meta-data; p. 12, paragraph 0108] for the application and generate a second dynamic component to replace the first dynamic component [That dynamically generates an interaction logic layer and customization which is then adapted at runtime; p. 3, paragraph 0029], wherein the second dynamic component is configured to function according to the new set of requirements [customization meta-data 54 and generates functional or customized presentations; p. 12, paragraph 0109 and p. 13, paragraph 0123].

Although Lucassen teaches the invention substantially, Lucassen does not specifically disclose determining whether the new set of requirements includes changes from the initial set of requirements and generating a second dynamic component to replace the first dynamic component if the new set of requirements includes changes from the initial set of requirements, wherein the dynamic component generator is configured to generate the second dynamic component to replace the first dynamic component in the application by modifying or overwriting the first dynamic component.

However, Green teaches a model-view-controller framework [p. 10, paragraph 0122], determining whether the new set of requirements includes changes from the initial set of requirements [As additional requirements arise 60, new software components 20 are created, existing software components 20 modified 62, 64, or a combination thereof; p. 5, paragraph 0060] and if the new set of requirements includes changes from the initial set of requirements [application requirements is determined 70, either manually, heuristically, automatically, or by any combination thereof; p. 12, paragraph 0150], generate a second dynamic component to replace the first dynamic component in the application [new software components 20 are created, existing software components 20 modified 62, 64, or a combination thereof; p. 5, paragraph

0060] wherein the dynamic component generator is configured to generate the second dynamic component to replace the first dynamic component [p. 5, paragraph 0060] by modifying [p. 5, paragraph 0064] or overwriting the first dynamic component [existing software components 20 modified 62, 64; p. 5, paragraph 0060].

It would have been obvious to a person of ordinary skill in the art at the time the invention was made to modify the invention of Lucassen to incorporate the features of determining whether the new set of requirements includes changes from the initial set of requirements and generating a second dynamic component to replace the first dynamic component if the new set of requirements includes changes from the initial set of requirements, and generate the second dynamic component to replace the first dynamic component by modifying or overwriting the first dynamic component because Green's teachings allow application development to drive changes to the architecture using a set of life cycle rules [p. 4, paragraph 0059 of Green] and allow software components that do not fit the current architecture to be restructured to ensure conformance while retaining the original intent of the requirement [p. 5, paragraph 0061 of Green].

8. As to claim 14, Lucassen as modified teaches a method [p. 2, paragraph 0017 of Lucassen], comprising:

installing one or more application modules [an application data layer 51, a business logic layer 52, an interaction logic layer 53 a customization layer 54, and application process 55; p. 11, paragraph 0105 of Lucassen] each comprising a static component [application data layer 51 comprises data content, file services and databases, and comprises all of the backend information; p. 11, paragraph 0105 of Lucassen];

one or more dynamic component generators [interaction manager 57; p. 12, paragraph 0109 of Lucassen] receiving an initial set of requirements for the application modules [meta-data; p. 12, paragraph 0108 of Lucassen];

the one or more dynamic component generators [interaction manager 57; p. 12, paragraph 0109 of Lucassen] generating one or more initial dynamic components for the one or more application modules [That dynamically generates an interaction logic layer and customization which is then adapted at runtime; p. 3, paragraph 0029 of Lucassen], wherein the one or more initial dynamic components are configured to function according to the initial set of requirements [customization meta-data 54 and generates functional or customized presentations; p. 12, paragraph 0109 and p. 13, paragraph 0123 of Lucassen];

receiving a new set of requirements for the application modules [meta-data; p. 12, paragraph 0108 of Lucassen];

determining whether the new set of requirements includes changes from the initial set of requirements [application requirements is determined 70, either manually, heuristically, automatically, or by any combination thereof; p. 12, paragraph 0150 of Green]; and

if the new set of requirements includes changes from the initial set of requirements, generating one or more new dynamic components to replace the one or more initial dynamic components in the application modules [As additional requirements arise 60, new software components 20 are created, existing software components 20 modified 62, 64, or a combination thereof; p. 5, paragraph 0060 of Green], wherein the one or more new dynamic components are configured to function according to the new set of requirements [p. 5, paragraph 0064 of Green], wherein said generating one or more new dynamic components comprises replacing the one or more initial dynamic components [p. 5, paragraph 0060 of Green] by the one or more new dynamic components by modifying [p. 5, paragraph 0064 of Green] or overwriting each of the one or more initial dynamic components [existing software components 20 modified 62, 64; p. 5, paragraph 0060 of Green].

9. As to claim 27, Lucassen as modified teaches a method [p. 2, paragraph 0017 of Lucassen], comprising:

installing one or more application modules [an application data layer 51, a business logic layer 52, an interaction logic layer 53 a customization layer 54, and application process 55; p. 11, paragraph 0105 of Lucassen], wherein at least a first one of the application modules comprises a first dynamic component [interaction logic layer 53; p. 11 – 12, paragraph 0107 of Lucassen] and a static component [application data layer 51 comprises data content, file services and databases, and comprises all of the backend information; p. 11, paragraph 0105 of Lucassen], wherein the first dynamic component and the static component are configured to function according to an initial set of requirements for the application [pp. 4 – 5, paragraph 0041 of Lucassen];

one or more dynamic component generators [interaction manager 57; p. 12, paragraph 0109 of Lucassen] receiving a new set of requirements for the application modules [meta-data; p. 12, paragraph 0108 of Lucassen];

the one or more dynamic component generators [interaction manager 57; p. 12, paragraph 0109 of Lucassen] determining whether the new set of requirements includes changes from the initial set of requirements [As additional requirements arise 60, new software components 20 are created, existing software components 20 modified 62, 64, or a combination thereof; p. 5, paragraph 0060 of Green]; and

if the new set of requirements includes changes from the initial set of requires [As additional requirements arise 60, new software components 20 are created, existing software components 20 modified 62, 64, or a combination thereof; p. 5, paragraph 0060 of Green], the one or more dynamic component generators generating a new dynamic component to replace the first dynamic component in the application modules [That dynamically generates an interaction logic layer and customization which is then adapted at runtime; p. 3, paragraph 0029 of Lucassen], wherein the new dynamic component is configured to function according to the new set of requirements [customization meta-data 54 and generates functional or customized presentations; p. 12, paragraph 0109 and p. 13, paragraph 0123 of Lucassen], wherein said generating comprises replacing the first dynamic component [p. 5, paragraph 0060 of Green] with the new dynamic component by modifying [p. 5, paragraph 0064 of Green] or overwriting the first dynamic component [existing software components 20 modified 62, 64; p. 5, paragraph 0060 of Green].

10. As to claim 41, this is a product claim that corresponds to system claim 1; note the rejection to claim 1 above, which also meet this product claim.

11. As to claim 2, Lucassen teaches the dynamic component generator does not change the static component in response to the new set of requirements [p. 11, paragraph 0105].

12. As to claim 3, Lucassen teaches the dynamic component generator is configured to generate a second dynamic component to replace the first dynamic component by modifying the first dynamic component in response to the new set of requirements [p. 15, paragraph 0146].

13. As to claim 4, Lucassen teaches the dynamic component generator is configured to replace the first dynamic component by overwriting the first dynamic component in the computer-accessible medium in response to the new set of requirements [interaction logic (some elements can be added, remove or replaced; p. 12, paragraph 0108].

14. As to claim 5, Lucassen teaches the new set of requirements is formatted according to an extensible Mark-up Language (XML) schema and stored in the

computer-accessible medium [data models 22 (or data type primitives) are XML Schema compliant; p. 6, 0059].

15. As to claim 6, Lucassen teaches the one or more application modules comprise a second application module comprising a static component and a dynamic component [p. 11, paragraph 0105].

16. As to claim 7, Lucassen teaches the dynamic component generator is configured to generate a new dynamic component [p. 3, paragraph 0029] for the second application module in response to receiving the new set of requirements [meta-data; p. 12, paragraph 0108].

17. As to claim 8, Lucassen teaches another dynamic component generator for the dynamic component of the second application module, wherein the other dynamic component generator is configured to generate a new dynamic component for the second application module [p. 3, paragraph 0029] in response to receiving a new set of requirements for the second application module [meta-data; p. 12, paragraph 0108].

18. As to claim 9, Lucassen as modified teaches the first application module is a controller module [Controllers C1, C2 and C3; p. 2, paragraph 0014 of Lucassen], wherein the static component as a router component configured to receive user input [composite business software component 20 is static; pp. 6 – 7, paragraph 0086 of

Green] and wherein the dynamic component is an application logic component coupled to the router component [interaction logic layer 53; p. 11 – 12, paragraph 0107 of Lucassen], wherein the application logic component is configured to function according to a current set of application requirements in response to the user input [customization meta-data 54 and generates functional or customized presentations; p. 12, paragraph 0109 and p. 13, paragraph 0123 of Lucassen].

19. As to claim 10, Lucassen teaches the application logic component comprises an Enterprise Java Bean (EJB) session bean [p. 11, paragraph 0095].

20. As to claim 11, Lucassen teaches the first application module is a model module [Model M; p. 5, paragraph 0044], wherein the static component is a static data model configured to function independent of an application data representation [p. 11, paragraph 0105], and wherein the dynamic component is a dynamic data model configured to function dependent upon the application data representation [p. 12, paragraph 0109 and p. 13, paragraph 0123] and according to a current set of application requirements [meta-data; p. 12, paragraph 0108] in response to the user input [p. 3, paragraph 0029].

21. As to claim 12, Lucassen teaches the dynamic data model comprises an Enterprise Java Bean (EJB) entity bean [p. 11, paragraph 0095].

22. As to claim 16, Lucassen teaches the generating one or more new dynamic components comprises replacing the one or more initial dynamic components by the one or more new dynamic components by modifying the each of the one or more initial dynamic components in response to the new set of requirements [p. 15, paragraph 0146].

23. As to claim 17, Lucassen teaches the generating one or more new dynamic components comprises replacing the one or more initial dynamic components by the one or more new dynamic components by overwriting each of the one or more initial dynamic components in a computer-accessible medium in response to the new set of requirements [p. 12, paragraph 0108].

24. As to claim 18, Lucassen teaches the generating is performed by one or more dynamic component generators, wherein the one or more dynamic component generators are comprised within the same application as the one or more application modules [p. 3, paragraph 0029].

25. As to claim 19, Lucassen teaches the generating is performed by one or more dynamic component generators comprised within an application server container, wherein the application modules are comprised within the same application server container [interaction components 91, 92 register with the container 92 and the contact

between the container 92 and components 90, 91 is programmed in the container 92; p. 15, paragraph 0151].

26. As to claim 20, Lucassen teaches the generating the static components comprised by the one or more application modules are not changed in response to the new set of requirements [p. 11, paragraph 0105].

27. As to claim 21, Lucassen teaches the new set of requirements is formatted according to an extensible Mark-up Language (XML) schema [data models 22 (or data type primitives) are XML Schema compliant; p. 6, 0059].

28. As to claim 22, Lucassen as modified teaches one of the one or more application modules is a controller module [Controllers C1, C2 and C3; p. 2, paragraph 0014 of Lucassen], wherein the static component is a router component configured to receive user input [composite business software component 20 is static; pp. 6 – 7, paragraph 0086 of Green], and wherein a dynamic component generated for the one of the one or more application modules is an application logic component [interaction logic layer 53; p. 11 – 12, paragraph 0107 of Lucassen] coupled to the router component [IGCRouter; p. 11, paragraph 0143 of Green], wherein the application logic component is configured to function according to a current set of requirements in response to the user input [customization meta-data 54 and generates functional or customized presentations; p. 12, paragraph 0109 and p. 13, paragraph 0123 of Lucassen].

29. As to claim 23, Lucassen teaches the application logic component comprises an Enterprise Java Bean (EJB) session bean [p. 11, paragraph 0095].

30. As to claim 24, Lucassen teaches one of the one or more application modules is a model module [Model M; p. 5, paragraph 0044], wherein the static component is a static data model configured to function independent of an application data representation [p. 11, paragraph 0105], and wherein a dynamic component generated for the one of the one or more application modules is a dynamic data model configured to function dependent upon the application data representation [p. 12, paragraph 0109 and p. 13, paragraph 0123] and according to a current set of requirements [meta-data; p. 12, paragraph 0108] in response to the user input [p. 3, paragraph 0029].

31. As to claim 25, Lucassen teaches the dynamic data model comprises an Enterprise Java Bean (EJB) entity bean [p. 11, paragraph 0095].

32. As to claim 28, Lucassen teaches the generating is performed by one or more dynamic component generators, wherein the one or more dynamic component generators are comprised within the same application server as the one or more application modules [p. 3, paragraph 0029].

33. As to claim 29, Lucassen teaches the generating is performed by one or more dynamic component generator comprised within an application server container, wherein the one or more application modules are comprised within the same application server container [interaction components 91, 92 register with the container 92 and the contact between the container 92 and components 90, 91 is programmed in the container 92; p. 15, paragraph 0151].

34. As to claim 30, Lucassen teaches in said generating, the static component does not change in response to the new set of requirements [p. 11, paragraph 0105].

35. As to claim 31, Lucassen teaches in said generating, the second dynamic component replaces the first dynamic component by modifying the first dynamic component in response to the new set of requirements [p. 15, paragraph 0146].

36. As to claim 32, Lucassen teaches in said generating, the second dynamic component replaces the first dynamic component by overwriting the first dynamic component in a computer-accessible medium in response to the new set of requirements [p. 12, paragraph 0108].

37. As to claim 33, Lucassen teaches wherein the new set of requirements is formatted according to an extensible Mark-up Language (XML) schema and stored in

the computer-accessible medium [data models 22 (or data type primitives) are XML Schema compliant; p. 6, 0059].

38. As to claim 34, Lucassen teaches wherein the one or more application modules comprise a second application module comprising a static component and a dynamic component [p. 11, paragraph 0105].

39. As to claim 35, Lucassen teaches generating a new dynamic component [p. 3, paragraph 0029] for the second application module in response to receiving the new set of requirements [meta-data; p. 12, paragraph 0108].

40. As to claims 36 and 37, these are similar in scope to claims 22 and 23; therefore, they are rejected for the same reasons as claims 22 and 23 above.

41. As to claims 38 – 39, these are similar in scope to claims 11 – 12; therefore, they are rejected for the same reasons as claims 11 – 12 above.

42. As to claims 42 – 48 and 51 – 52, these are product claims that correspond to system claims 2 – 8 and 11 – 12; note the rejections to claims 2 – 8 and 11 – 12 above, which also meet these product claims.

43. As to claims 49 – 50, these are product claims that correspond to system claims 9 and 10; note the rejection to claims 9 and 10 above, which also meet these product claims.

Conclusion

44. **THIS ACTION IS MADE FINAL.** Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the mailing date of this final action.

CONTACT INFORMATION

45. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Li B. Zhen whose telephone number is (571) 272-3768. The examiner can normally be reached on Mon - Fri, 8:30am - 5pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William Thomson can be reached on 571-272-3718. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Li B. Zhen
Examiner
Art Unit 2194

LBZ

[illegible] 8/19/2007